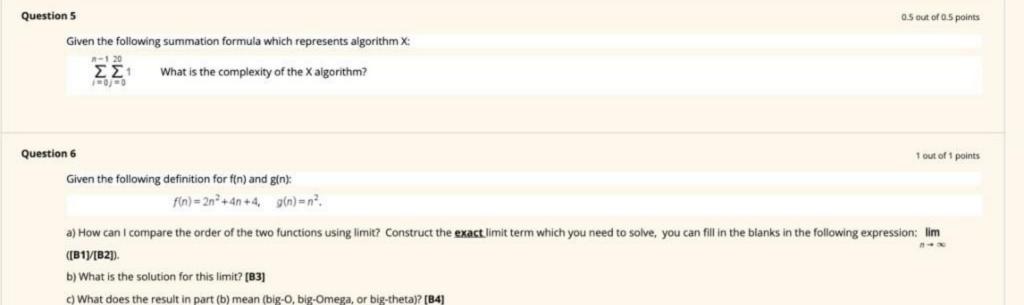
Question 1	Needs Gradi
Use the formal definition to prove/disprove the following: $(2n + 5)^2 = \theta(n^2)$.	
You must show all solution steps.	
Question 2	1000 P.O.
Use the formal definition to prove or disprove the following: $3n^2 + 6n = \Omega(n^2 \log n)$.	Needs Grad
You must show all solution steps.	
Suppose we are sorting an array of the following integers [8, 5, 7, 1, 9, 3], and we have just finished the first iteration with the array looking like this: [1, 5, 7, 8, 9, 3]. algorithm would be:	0 out of 0.25 poi
Question 4	0.25 out of 1 poi
Given the following definition for f(n) and g(n):	
$f(n) = 2n^2 + 4n + 4$, $g(n) = n^2$.	
pick the correct values for c and n0 that will prove the following relation:	
pick the correct values for c and n0 that will prove the following relation: $f(n) = \Omega(g(n)).$	



Give a big-Oh characterization, in terms of n,of the running time for each of the following algorithms (use the drop-down):

```
Question
public woid funcl(int n) (
  for (int i = n; i > 0; i--) {
     System.out.println(i);
     for (int j = 0; j < i; j++)
        System.out.println(j);
  System.out.println("Goodbye!");
public void func2(int n) (
  for (int m=1; m <= n; m++) {
     system.out.println(m);
     1 - n;
     while (i > 0 ) {
        system.out.println(i);
        i = i / 2;
public void func3(int [] A) {
   int L = A.length; // length is a variable which contains
                     // the number of elements in an array
   if (L%2 ==0)
       System.out.println("Even elements");
   else
       System.out.println("Odd elements");
public void func4(int n, String msg) {
  for (int i = n; i > 0; i--)
      for (int j = 0; j < 1; j++)
          for (int k = 0; k < 1000; k++)
              System.out.println(msg);
public void func5(LinkedList list) (
   int L = list.size(); // size() is a method which iterates
                        // through a linked list and returns
                        // the number of elements in the list
  if (L%2 ==0)
      System.out.println("Even elements");
  else
      System.out.println("Odd elements");
```



Consider the following algorithm:

ALGORITHM Mystery(n)

//Input: A nonnegative integer
$$n$$
 $S \leftarrow 0$

for $i \leftarrow 1$ to n do
 $S \leftarrow S + i * i$

return S

- a. What does this algorithm compute? [B1]
- b. What is its basic operation? [B2]
- c. How many times is the basic operation executed? [B3]

Specified Answer for: B1 3 The sum of squares of n numbers

Specified Answer for: B2 6 Multiplication

Specified Answer for: B3 ON times

Correct Answers for: B1		
Evaluation Method	Correct Answer	Case Sensitivity
📀 Exact Match	sums the square of number from 1 to n	rs
Correct Answers for: B2		
Evaluation Method	Correct Answer	Case Sensitivity
🖔 Contains	addition	
Contains Contains	multiplication	
Correct Answers for: B3		
Evaluation Method	Correct Answer	Case Sensitivity
Contains	n	

Mark each of the following statements with either (1) True or (2) False:

	Question	Correct Match	Selected Match
•	The base of a log changes the growth of the logarithm function by a constant.	🤨 1. True	2. False
	It is not possible to classify algorithms according to an underlying design idea, algorithms are only classified based on their complexity.	2. False	2. False
	It is possible that $f(n) = O(g(n))$ and $g(n) = O(f(n))$.	🤨 1. True	🐧 1. True
	The Traveling Salesman Problem is a famous example of a graph problem.	⊙ 1. True	🤨 1. True
	Combinatorial problems are the most difficult problems in computing.	🤨 1. True	🤨 1. True
	Shortest-path problem (what is the best route between two cities?) is an example of a combinatorial problem which can be solved by an efficient algorithm.	🤨 1. True	🤨 1. True

Given the function $f(n) = \sqrt{n}$,

Identify the asymptotic relationship between f(n) and g(n) functions by picking the **most accurate** asymptotic relation to fill in the blank in the following equation (i.e, replace the ? with an appropriate asymptotic symbol from the provided choices):

$$f(n) = _?_(g(n)).$$

Question Correct Match Selected Match
$$g(n) = \log \sqrt{n}$$
 \odot C. Ω \odot B. θ \odot B. θ